APPENDIX A

```
/*
    DMXPlayerFrame.java

    Title:          DMXPlayer
    Author          Schemel
    Description:    This applet reads data from a text file and writes it to a DMX channel...
*/

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.*;
import java.io.*;
import com.colorkinetics.control.dmx.*;

public class DMXPlayerFrame extends JFrame
{

// IMPORTANT: Source code between BEGIN/END comment pair will be regenerated
// every time the form is saved. All manual changes will be overwritten.
// BEGIN GENERATED CODE
    // member declarations
    javax.swing.JButton loadFileButton = new javax.swing.JButton();
    javax.swing.JLabel fileNameStatic = new javax.swing.JLabel();
    javax.swing.JLabel numEntriesStatic = new javax.swing.JLabel();
    javax.swing.JLabel minStatic = new javax.swing.JLabel();
    javax.swing.JLabel maximumStatic = new javax.swing.JLabel();
    javax.swing.JLabel delayStatic = new javax.swing.JLabel();
    javax.swing.JLabel fileNameLabel = new javax.swing.JLabel();
    javax.swing.JLabel numEntriesLabel = new javax.swing.JLabel();
    javax.swing.JLabel minLabel = new javax.swing.JLabel();
    javax.swing.JLabel maxLabel = new javax.swing.JLabel();
    javax.swing.JButton startButton = new javax.swing.JButton();
    javax.swing.JButton stopButton = new javax.swing.JButton();
    javax.swing.JButton exitButton = new javax.swing.JButton();
    javax.swing.JButton initButton = new javax.swing.JButton();
// END GENERATED CODE
    IntegerInput delayInput = new IntegerInput(100,6,0,1000000);

    LinkedList dataList = new LinkedList();
    int dataCount = 0;
    Double dataMax = new Double(0.0);
    Double dataMin = new Double(0.0);

    Runnable playerThread;
    Thread th;

    private DMXUniverse DMX;
    private String[] interNames;
    private int selectedInterface = -1;

    public DMXPlayerFrame()
    {
        DMX = new DMXUniverse("DMXPlayer");
    }

    public void initComponents() throws Exception
    {
// IMPORTANT: Source code between BEGIN/END comment pair will be regenerated
// every time the form is saved. All manual changes will be overwritten.
// BEGIN GENERATED CODE
        // the following code sets the frame's initial state
        loadFileButton.setSize(new java.awt.Dimension(110, 30));
        loadFileButton.setVisible(true);
        loadFileButton.setText("Load File...");
        loadFileButton.setLocation(new java.awt.Point(110, 70));

        fileNameStatic.setSize(new java.awt.Dimension(120, 20));
        fileNameStatic.setVisible(true);
        fileNameStatic.setText("File Name:");
        fileNameStatic.setLocation(new java.awt.Point(30, 150));

        numEntriesStatic.setSize(new java.awt.Dimension(120, 20));
```

```
numEntriesStatic.setVisible(true);
numEntriesStatic.setText("Number of Entries:");
numEntriesStatic.setLocation(new java.awt.Point(30, 190));

minStatic.setSize(new java.awt.Dimension(120, 20));
minStatic.setVisible(true);
minStatic.setText("Minimum:");
minStatic.setLocation(new java.awt.Point(30, 220));

maximumStatic.setSize(new java.awt.Dimension(120, 20));
maximumStatic.setVisible(true);
maximumStatic.setText("Maximum:");
maximumStatic.setLocation(new java.awt.Point(30, 250));

delayStatic.setSize(new java.awt.Dimension(120, 20));
delayStatic.setVisible(true);
delayStatic.setText("Delay(ms):");
delayStatic.setLocation(new java.awt.Point(30, 300));

fileNameLabel.setSize(new java.awt.Dimension(180, 20));
fileNameLabel.setVisible(true);
fileNameLabel.setText("None Selected");
fileNameLabel.setLocation(new java.awt.Point(150, 150));

numEntriesLabel.setSize(new java.awt.Dimension(180, 20));
numEntriesLabel.setVisible(true);
numEntriesLabel.setText("-");
numEntriesLabel.setLocation(new java.awt.Point(150, 190));

minLabel.setSize(new java.awt.Dimension(180, 20));
minLabel.setVisible(true);
minLabel.setText("-");
minLabel.setLocation(new java.awt.Point(150, 220));

maxLabel.setSize(new java.awt.Dimension(180, 20));
maxLabel.setVisible(true);
maxLabel.setText("-");
maxLabel.setLocation(new java.awt.Point(150, 250));

startButton.setSize(new java.awt.Dimension(90, 30));
startButton.setVisible(true);
startButton.setText("Start");
startButton.setLocation(new java.awt.Point(20, 380));

stopButton.setSize(new java.awt.Dimension(90, 30));
stopButton.setVisible(true);
stopButton.setText("Stop");
stopButton.setLocation(new java.awt.Point(120, 380));

exitButton.setSize(new java.awt.Dimension(90, 30));
exitButton.setVisible(true);
exitButton.setText("Exit");
exitButton.setLocation(new java.awt.Point(240, 380));

initButton.setSize(new java.awt.Dimension(110, 30));
initButton.setVisible(true);
initButton.setText("Initialize...");
initButton.setLocation(new java.awt.Point(110, 20));

setSize(new java.awt.Dimension(353, 451));
getContentPane().setLayout(null);
setTitle("DMXPlayer");
setLocation(new java.awt.Point(0, 0));
getContentPane().add(loadFileButton);
getContentPane().add(fileNameStatic);
getContentPane().add(numEntriesStatic);
getContentPane().add(minStatic);
getContentPane().add(maximumStatic);
getContentPane().add(delayStatic);
getContentPane().add(fileNameLabel);
getContentPane().add(numEntriesLabel);
getContentPane().add(minLabel);
getContentPane().add(maxLabel);
getContentPane().add(startButton);
```

```
        getContentPane().add(stopButton);
        getContentPane().add(exitButton);
        getContentPane().add(initButton);

        loadFileButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent e) {
                loadFileButtonActionPerformed(e);
            }
        });
        startButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent e) {
                startButtonActionPerformed(e);
            }
        });
        stopButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent e) {
                stopButtonActionPerformed(e);
            }
        });
        exitButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent e) {
                exitButtonActionPerformed(e);
            }
        });
        initButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent e) {
                initButtonActionPerformed(e);
            }
        });
        addWindowListener(new java.awt.event.WindowAdapter() {
            public void windowClosing(java.awt.event.WindowEvent e) {
                thisWindowClosing(e);
            }
        });
// END GENERATED CODE

        delayInput.setSize(new java.awt.Dimension(90, 30));
        delayInput.setVisible(true);
        delayInput.setLocation(new java.awt.Point(250, 200));
        getContentPane().add(delayInput);
    }

    private boolean mShown = false;

    public void addNotify()
    {
        super.addNotify();

        if (mShown)
            return;

        // resize frame to account for menubar
        JMenuBar jMenuBar = getJMenuBar();
        if (jMenuBar != null) {
            int jMenuBarHeight = jMenuBar.getPreferredSize().height;
            Dimension dimension = getSize();
            dimension.height += jMenuBarHeight;
            setSize(dimension);
        }

        mShown = true;
    }

    // Close the window when the close box is clicked
    void thisWindowClosing(java.awt.event.WindowEvent e)
    {
        setVisible(false);
        dispose();
        System.exit(0);
    }

    public void exitButtonActionPerformed(java.awt.event.ActionEvent e)
    {
```

```
        setVisible(false);
        dispose();
        System.exit(0);
    }

    public void startButtonActionPerformed(java.awt.event.ActionEvent e)
    {
        ((DMXPlayerThread)playerThread).delay = delayInput.getValue();
        ((DMXPlayerThread)playerThread).shouldRun = true;
//      th.setPriority(th.MAX_PRIORITY);
    }

    public void stopButtonActionPerformed(java.awt.event.ActionEvent e)
    {
        ((DMXPlayerThread)playerThread).shouldRun = false;
//      th.setPriority(th.MIN_PRIORITY);
    }

    public void loadFileButtonActionPerformed(java.awt.event.ActionEvent e)
    {
        FileDialog d = new FileDialog(this, "Load data file", FileDialog.LOAD);

        d.show();

        String fileName = d.getFile();
        String dirName = d.getDirectory();

        if(fileName != null) {
            try {
                fileNameLabel.setText(fileName);
                File f = new File(dirName, fileName);
                FileReader fr = new FileReader(f);

                int size = (int)f.length();
                char[] data = new char[size];
                int chars_read = 0;

                while(chars_read < size)
                    chars_read += fr.read(data, chars_read, size - chars_read);

                fr.close();

                String s = new String(data);

                StringTokenizer st = new StringTokenizer(s);

                String t = new String();
                while(st.hasMoreTokens()) {
                    t = st.nextToken();
                    try {
                        Double val = new Double(t);
                        dataList.add(val);
                    } catch (NumberFormatException nfe) {}
                }
            } catch (IOException ioe) {
                JOptionPane errorDialog = new JOptionPane();
                errorDialog.showMessageDialog(null, "Caught an IOException in LoadFileActionPerformed.");
            }

            dataCount = dataList.size();
            numEntriesLabel.setText(String.valueOf(dataCount));

            if(dataCount > 0) {
                Double D = (Double)dataList.get(0);
                dataMax = D;
                dataMin = D;
                for(int i = 1; i < dataCount; i++) {
                    D = (Double)dataList.get(i);
                    Double newMax = new Double(Math.max(dataMax.doubleValue(), D.doubleValue()));
                    dataMax = newMax;
                    Double newMin = new Double(Math.min(dataMin.doubleValue(), D.doubleValue()));
                    dataMin = newMin;
                }
            }
```

```
            maxLabel.setText(String.valueOf(dataMax));
            minLabel.setText(String.valueOf(dataMin));

        }

        playerThread = new DMXPlayerThread(delayInput.getValue(), dataList, dataMin, dataMax, DMX);
        th = new Thread(playerThread);
        th.setPriority(th.MIN_PRIORITY);
        th.start();
    }

    public void initButtonActionPerformed(java.awt.event.ActionEvent e)
    {
        if (DMXUniverse.isAvailable()) {
            interNames = DMXUniverse.getInterfaceNames();
            String reply = (String)JOptionPane.showInputDialog(
                this,
                "Select a DMX  nterface:",
                "Select",
                JOptionPane.QUESTION_MESSAGE,
                null,
                interNames,
                interNames[0]);

            if (reply == null) return;

            selectedInterface = -1;
            for (int i = 0; i < interNames.length; i++) {
                if (reply == interNames[i]) {
                    selectedInterface = i;
                    break;
                }
            }

            try {
                DMX.close();
                DMX.selectInterfaceFromIndex(selectedInterface);
                DMX.setReadyLive();
                System.out.println("Selecting interface number " + selectedInterface);
            }
            catch (DMXException x) {
                JOptionPane.showMessageDialog(this,
                    "DMX Error\n" + x.getMessage(),
                    "DMX Error",
                    JOptionPane.ERROR_MESSAGE);
                selectedInterface = -1;
            }

        } else {
            JOptionPane.showMessageDialog(this,
                "Can't load DMX library!",
                "DMX Error",
                JOptionPane.ERROR_MESSAGE);
            selectedInterface = -1;
        }
    }

}
```

```
            maxLabel.setText(String.valueOf(dataMax));
            minLabel.setText(String.valueOf(dataMin));

        }

        playerThread = new DMXPlayerThread(delayInput.getValue(), dataList, dataMin, dataMax, DMX);
        th = new Thread(playerThread);
        th.setPriority(th.MIN_PRIORITY);
        th.start();
    }

    public void initButtonActionPerformed(java.awt.event.ActionEvent e)
    {
        if (DMXUniverse.isAvailable()) {
            interNames = DMXUniverse.getInterfaceNames();
            String reply = (String)JOptionPane.showInputDialog(
                this,
                "Select a DMX interface:",
                "Select",
                JOptionPane.QUESTION_MESSAGE,
                null,
                interNames,
                interNames[0]);

            if (reply == null) return;

            selectedInterface = -1;
            for (int i = 0; i < interNames.length; i++) {
                if (reply == interNames[i]) {
                    selectedInterface = i;
                    break;
                }
            }

            try {
                DMX.close();
                DMX.selectInterfaceFromIndex(selectedInterface);
                DMX.setReadyLive();
                System.out.println("Selecting interface number " + selectedInterface);
            }
            catch (DMXException x) {
                JOptionPane.showMessageDialog(this,
                    "DMX Error\n" + x.getMessage(),
                    "DMX Error",
                    JOptionPane.ERROR_MESSAGE);
                selectedInterface = -1;
            }

        } else {
            JOptionPane.showMessageDialog(this,
                "Can't load DMX library!",
                "DMX Error",
                JOptionPane.ERROR_MESSAGE);
            selectedInterface = -1;
        }
    }

}
```

```
/*
    DMXPlayerThread.java

    Title:          DMXPlayer
    Author:         bchemel
    Description:    This application reads data from a text file and writes it to a DMX channel...
*/

import java.util.*;
import com.colorkinetics.control.dmx.*;
import javax.swing.*;
import java.awt.*;


public class DMXPlayerThread implements Runnable {
    public int delay;
    public Double max;
    public Double min;
    public double oldVal = 0.0;
    public double mean = 0.0,
    public LinkedList data;
    public boolean shouldRun = false;

    public DMXUniverse DMX;
    private String[] interNames;
    private int selectedInterface = -1;
    private JFrame frame;

    private static final int IRDA_BAUD = 1;
    private static final int SP_BAUD_SJ1 = 10;
    private static final int SP_BAUD_SJ2 = 4;
    private static final int SJ1 = 1;
    private static final int SJ2 = 2;

    private int irdaBaud = IRDA_BAUD;
    private int spBaud = SP_BAUD_SJ2;
    private int interfaceValue = SJ2;

    public DMXPlayerThread(int d, LinkedList l, Double Min, Double Max, DMXUniverse univ) {
        DMX = univ;
        delay = d;
        data = l;
        max = Max;
        min = Min;
        int len = data.size();

        for(int i = 0; i < len; i++) {
            mean += ((Double)data.get(i)).doubleValue() / (double)len;
        }
    }

    public void run() {
        while(true) {
            while(shouldRun) {
                DMXFrame buffer = new DMXFrame(512);

                try{
                    Double D = (Double)data.removeFirst();
                    System.out.println("New Value: " + D);
                    data.add(D);
                    float span = (float)max.doubleValue() - (float)min.doubleValue();
//                  float hue = ((float)D.doubleValue() - (float)min.doubleValue())/span;
                    double d = D.doubleValue();

                    try {
                        for(int i = 0; i < delay; i++) {
                            double outVal = oldVal + (((double)i + 1.0)/(double)delay)*(d - oldVal);
                            float hue = (float)0.0;
                            if(outVal > mean) {
                                hue = (float)(1.0/3.0);
                            } else {
                                hue = (float)0.0;
                            }
                            float brightness = (float)0.00 + (Math.abs((float)(outVal - mean))/span);
```

```
                    Color c = new Color(Color.HSBtoRGB(hue, (float)1.0, brightness));

                    buffer.setData(0, c.getRed(), 0);
                    buffer.setData(1, c.getGreen(), 0);
                    buffer.setData(2, c.getBlue(), 0);
                    DMX.sendFrame(buffer);
                    buffer.resetPriority();
//                  Thread.sleep(1);
                }
            } catch (Exception x) {
                System.out.println("Caught a DMX error!");
                System.out.println(x);
            }

            oldVal = d;
        } catch (Exception e) {}
    }
  }
 }
}
```

```
// IntegerInput.java

//
//
//
//
//
//

import javax.swing.*;
import javax.swing.text.*;
import java.awt.*;
import java.awt.event.*;

/**
 * An input field for integers
 *
 * @version 1.0
 * @author Mike Blackwell
 */
public class IntegerInput extends JTextField {

    private int minVal, maxVal;

    /**
     * Create a new integer input field with an initial value of <code>initval</code>,
     * and a column width of <code>columns</code>. Minimum and maximum values are
     * specified.
     */
    public IntegerInput(int initval, int columns, int min, int max) {
        super(columns);
        minVal = min;
        maxVal = max;
        setDocument(new IntegerDocument());
        if (initval < min) {
            setValue(min);
        } else if (initval > max) {
            setValue(max);
        } else {
            setValue(initval);
        }
    }


    /**
     * Create a new integer input field with a min of 0 and a max of
     * Integer.MAX_VALUE.
     */
    public IntegerInput(int initval, int columns) {
        this(initval, columns, 0, Integer.MAX_VALUE);
    }


    /**
     * Return the integer value contained in the field.
     */
    public int getValue() {
        int val;
        try {
            val = Integer.parseInt(getText());
        }
        catch (NumberFormatException e) {
            val = 0;
        }
        return val;
    }


    /**
     * Set the integer value of the field.
     */
    public void setValue(int val) {
        setText(Integer.toString(val));
    }
```

```
    public int getMinValue() { return minVal; }

    public int getMaxValue() { return maxVal; }

    public void setMinValue(int v) { minVal = v; }

    public void setMaxValue(int v) { maxVal = v; }


    class IntegerDocument extends PlainDocument {
        public void insertString(int offset, String s, AttributeSet attrs)
                throws BadLocationException {

            // Test build the resulting string first. Beep and don't
            // allow if it's not an integer, or if it's out of bounds.
            StringBuffer t = new StringBuffer(getText(0, getLength()));
            t.insert(offset, s);
            try {
                int v = Integer.parseInt(t.toString());
                if ((v < minVal) || (v > maxVal))
                    throw new Exception("Integer out of bounds");
            }
            catch (Exception e) {
                Toolkit.getDefaultToolkit().beep();
                return;
            }
            super.insertString(offset, s, attrs);
        }
    }
}
```

```
/*
    DMXPlayerApp.java

    Title:          DMXPlayer
    Author:         bchemel
    Description:    This application reads data from a text file and writes it to a DMX channel...
*/


import DMXPlayerFrame;
import javax.swing.*;

public class DMXPlayerApp
{
    public DMXPlayerApp()
    {
        try {
            // For native Look and Feel, uncomment the following code.
            /*
            try {
                UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
            }
            catch (Exception e) {
            }
            */
            DMXPlayerFrame frame = new DMXPlayerFrame();
            frame.initComponents();
            frame.setVisible(true);
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }

    // Main entry point
    static public void main(String[] args)
    {
        new DMXPlayerApp();
    }

}
```